What is claimed is:

1. A method for distributed computing, comprising:

sending from a server to a task processing module, a request to process a task;

receiving the task at the task processing module;

decomposing the task into a plurality of subtasks;

returning the subtasks to the server;

distributing the subtasks from the server to processors;

receiving the subtasks at the processors;

determining at the processors if code resides at the processors to process the subtasks received;

obtaining at the processors the code from a code source when the code does not exist at the processors;

determining at the processors if data exists at the processors for the subtasks received;

obtaining at the processors the data from a data source when the data does not exist at the processors;

executing at the processors the code to obtain results for the subtasks;

notifying the server that the results for the subtasks are obtained;

combining the results of the subtasks to obtain a task result.

2. The method of claim 1, comprising maintaining updated versions of at least one of system parameters, processing code and system operation code at the task processing module.

3. The method of claim 2, wherein maintaining comprises updating system parameters taken from a list including server addresses, server operating system identification, server

organizational type, server task purpose, processor operating system identification and server processing requirements.

4. The method of claim 3, wherein receiving the subtasks at the processors comprises checking system parameters to determine if the server is an approved server.

5. The method of claim 1, wherein sending the request to process a task comprises forming a dynamically linked library having links to at least one of processing code, code sources, data, data sources and results storage files.

6. The method of claim 5, comprising:

defining configuration sets for tasks to be requested by the server; and

incorporating in the dynamically linked library one of the configuration sets corresponding to the task in the request to process a task.

7. The method of claim 6, wherein defining the configuration sets comprises identifying at least one of subtask processing limits, boundary limits for the data, iteration limits and a number of processors desired for processing.

8. The method of claim 7, wherein combining comprises iteratively sending requests to process the task results based on the iteration limits.

9. The method of claim 6, wherein defining the configuration sets comprises maintaining a list of processors available to execute the code to obtain the results for the subtasks.

10. The method of claim 9, wherein maintaining the list comprises adding to the list processors for which availability signals are received and removing from the list processors for which availability signals are not been received within a predetermined period.

11. The method of claim 1, wherein returning the subtasks comprises compressing files corresponding to the subtasks.

12. The method of claim 11, wherein executing comprises compressing files corresponding to the results for the subtasks.

13. The method of claim 1, wherein executing comprises compressing files corresponding to the results for the subtasks.

14. The method of claim 1, comprising maintaining a list of processors available to execute the code to obtain the results for the subtasks.

15. The method of claim 14, wherein maintaining the list comprises adding to the list processors for which availability signals are received and removing from the list processors for which availability signals are not been received within a predetermined period.

16. The method of claim 1, wherein distributing comprises:

monitoring the processors; and

redistributing the subtasks when executing at the processors is delayed.

17. The method of claim 16, wherein monitoring comprises:

accessing the server from a remote site; and

initiating a browser application within the server, the browser application providing remote monitoring functionality.

18. The method of claim 1, wherein combining comprises iteratively sending requests to process the task results.

19. A distributed computing system, comprising:

a server module adapted to request processing of a task;

a processing module adapted to receive the task, decompose the task into a plurality of subtasks and return the subtasks to the server;

helper modules adapted to receive the subtasks distributed by the server, to obtain processing code and data to process the subtasks and return subtask results to the server, wherein the subtask results are combined to obtain a task result.

20. The system of claim 19, wherein the server module, the processing module and the helper modules are connected via a network.

21. The system of claim 20, wherein the network is one of an internet, an intranet, a local area network and a wide area network.

22. The system of claim 19, wherein the processing module is adapted to maintain at least one of updated system parameters, processing code and system operation code.

23. The system of claim 22, wherein the updated system parameters comprise at least one of server module addresses, server module operating system identification, server module organizational type, server module task purpose, helper module operating system identification and server module processing requirements.

24. The method of claim 23, wherein the helper modules verify the system parameters for the server module to determine if the server module is an approved server module.

25. The system of claim 19, comprising a dynamically linked library formed by the server module and adapted to provide links to at least one of processing code, data and subtask results storage files.

26. The system of claim 25, wherein the dynamically linked library comprises configuration information for the processing module and helper modules.

27. The system of claim 26, wherein the configuration information comprises at least one of subtask processing limits, boundary limits for the data, iteration limits and a number of helpers desired for processing.

28. The system of claim 27, comprising an iterative module adapted to iteratively request processing the task results based on the iteration limits.

29. The system of claim 26, wherein the configuration information comprises a helper module list of helper modules for which an availability signal has been received.

30. The system of claim 19, wherein the processing module comprises a subtask compression module adapted to return the subtasks in a compressed format.

31. The system of claim 30, wherein the helper modules comprise a results compression module adapted to return the subtask results in a compressed format.

32. The system of claim 19, wherein the helper modules comprise a results compression module adapted to return the subtask results in a compressed format.

33. The system of claim 19, comprising a helper module list of helper modules available to receive, process and return results for the subtasks.

34. The system of claim 33, wherein the helper modules initiate periodic availability signals to update the helper module list, whereby helper modules for which availability signals are received are added to the helper module list and helper modules for which availability signals are not received are removed from the helper module list.

35. The system of claim 19, wherein the server module comprises a monitoring module adapted to monitor the helper modules and redistribute the subtasks when at least one of the helper modules is delayed.

36. The method of claim 35, wherein the monitoring module comprises a browser application for accessing the server from a remote site and monitoring the helper modules from the remote site through the browser application.

37. The method of claim 19, comprising a browser application adapted to access the server from a remote site and operate the system from the remote site.

38. The method of claim 19, comprising an iterative module adapted to iteratively request processing the task results.

39. A method for distributed computing, comprising:

decomposing a task into a plurality of subtasks;

distributing the subtasks to processors;

determining at the processors if processing code exists at the processors to process the subtasks received;

obtaining at the processors the processing code from a code source when the code does not exist at the processors;

executing at the processors the processing code to obtain results for the subtasks;

combining the results of the subtasks to obtain a task result.

40. The method of claim 39, comprising maintaining updates of the processing code at the code source.

41. The method of claim 39, comprising forming a dynamically linked library to provide links to at least one of processing code, code sources and storage files for results of the subtasks.

42. The method of claim 39, wherein decomposing comprises compressing files corresponding to the subtasks.

43. The method of claim 42, wherein executing comprises compressing files corresponding to the results for the subtasks.

44. The method of claim 39, wherein executing comprises compressing files corresponding to the results for the subtasks.

45. A method for distributed computing, comprising:

decomposing a task into a plurality of subtasks;

distributing the subtasks to processors;

determining at the processors if data exists at the processors for the subtasks received;

obtaining at the processors the data from a data source when the data does not exist at the processors;

executing at the processors the subtasks using the data to obtain results for the subtasks;

combining the results of the subtasks to obtain the task result.

46. The method of claim 45, comprising forming a dynamically linked library to provide links to at least one of data, data sources and storage files for results of the subtasks.

47. The method of claim 45, wherein decomposing comprises compressing files corresponding to the subtasks.

48. The method of claims 47, wherein executing comprises compressing files corresponding to the results for the subtasks.

49. The method of claims 45, wherein executing comprises compressing files corresponding to the results for the subtasks.

50. A computer program tangibly stored on a computer-readable medium and operable to cause a computer to enable distributed computing of a task, the computer program comprising instructions to:

send a request to process the task from a server to a task processing module;

decompose the task into a plurality of subtasks;

distribute the subtasks to processors;

determine if code exists at the processors to process the subtasks;

obtain the code from a code source when the code does not exist at the processors;

determine if data exists at the processors for the subtasks;

obtain the data from a data source when the data does not exist at the processors;

execute the code to obtain results for the subtasks; and

combine the results of the subtasks to obtain a task result.

51. The computer program of claim 50, comprising instructions to maintain updated versions of at least one of system parameters, processing code and system operation code at the task processing module.

52. The computer program of claim 51, wherein the instructions to maintain comprise instructions to update system parameters taken from a list including server addresses, server operating system identification, server organizational type, server task purpose, processor operating system identification and server processing requirements.

53. The computer program of claim 52, comprising instructions to check system parameters to determine if the server is an approved server.

54. The computer program of claim 50, wherein the instructions to send the request to process a task comprise instructions to form a dynamically linked library having links to at least one of processing code, code sources, data, data sources and results storage files.

55. The computer program of claim 54, comprising instructions to:

define configuration sets for tasks to be requested by the server; and

incorporate in the dynamically linked library one of the configuration sets corresponding to the task in the request to process a task.

56. The computer program of claim 55, wherein the instructions to define the configuration sets comprise instructions to identify at least one of subtask processing limits, boundary limits for the data, iteration limits and a number of processors desired for processing.

57. The computer program of claim 56, wherein the instructions to combine comprise instructions to iteratively send requests to process the task results based on the iteration limits.

58. The computer program of claim 55, wherein the instructions to define the configuration sets comprise instructions to maintain a list of processors available to execute the code to obtain the results for the subtasks.

59. The computer program of claim 58, wherein the instructions to maintain the list comprise instructions to add to the list processors for which availability signals are received and instructions to remove from the list processors for which availability signals are not been received within a predetermined period.

60. The computer program of claim 50, wherein the instructions to decompose comprise instructions to compress files corresponding to the subtasks.

61. The computer program of claim 60, wherein the instructions to execute comprise instructions to compress files corresponding to the results for the subtasks.

62. The computer program of claim 50, wherein the instructions to execute comprise instructions to compress files corresponding to the results for the subtasks.

63. The computer program of claim 50, comprising instructions to maintain a list of processors available to execute the code to obtain the results for the subtasks.

64. The computer program of claim 63, wherein the instructions to maintain the list comprise instructions to:

add to the list processors for which availability signals are received; and

remove from the list processors for which availability signals are not been received within a predetermined period.

65. The computer program of claim 50, wherein the instructions to distribute comprise instructions to:

monitor the processors; and

redistribute the subtasks when results of one of the subtasks is delayed.

66. The computer program of claim 65, wherein the instructions to monitor comprise instructions to:

access the server from a remote site; and

initiate a browser application within the server, the browser application providing remote monitoring functionality.

67. The computer program of claim 50, wherein the instructions to combine comprise instructions to iteratively send requests to process the task results.